AMENDMENTS TO THE SPECIFICATION

In the Specification:

Please replace the paragraph beginning at pg. 1, ln. 5 with the following paragraph:

The present invention relates generally to computer systems, and more particularly to a system and method that employs file property handlers to facilitate compatibility between unstructured file property storage in byte streams and structured object representations of the file *via* promotion and demotion of file properties. The term item employed herein generally refers to a structured, schematized object that is stored in a structured object store. A file-backed item [[to]] refers to a structured object representation of the file in an object store. The term file can be used to represent an unstructured byte stream that corresponds to a given file-backed item.

Please replace the paragraph beginning at pg. 5, ln. 20 with the following paragraph:

If a file were to be modified, saved, and/or manipulated as an unstructured file, the bridge module or file property manager 130 serves to direct the transformation from the unstructured file to the structured object. The transformation is performed by the bridge module 130 invoking the file property handler 140 which would in turn perform ~~performs~~ a promotion operation to transform unstructured properties of an unmanaged file to structured properties associated with applications that operate against the structured store 160. If the promoted object were to be manipulated from the structured store application, then the file property handler 140 performs a demotion operation which causes properties to be reverse transformed into properties that are then updated in the unstructured file.

Please replace the paragraph beginning at pg. 9, ln. 8 with the following paragraph:

As illustrated in Fig. 2, if an application working with unstructured files 200 modifies a file 240 in the structured store namespace at 230, the FPH 210 performs an extraction and transformation of unstructured properties at 250, before returning the promoted item to be saved at 230. In contrast, demotion is generally invoked when a file-backed item is updated through a structured store API 200. The structured store API 200 allows applications to modify one or more file-backed items or parts of such items. After performing this, when the application tries to save the changed item(s) using the structured store API, the method in the structured store API that is invoked to do the save performs the following: [[()]] The method in the structured store API that performs the function of saving the item is conceptually referred to here as the 'save' method, although the exact method name is implementation-dependent.

Please replace the paragraph beginning at pg. 9, ln. 19 with the following paragraph:

If a file-backed item, or a part of it is modified through the structured store API [[260]], the structured store API's save method looks up the demoter corresponding to the file-backed item (based on file extension) and invokes it. The demoter takes in an item (or part of it) for read-write and a file stream for write and updates the file content based on the changes to the item. Demotion is a generally a synchronous operation since it is invoked during the operation that tries to save the item to the structured store. The save method described above updates the file 240 by invoking the demoter as well as write suitable properties to the item in the structured data store.

Please replace the paragraph beginning at pg. 10, ln. 21 with the following paragraph:

After 350, the process proceeds to 410 of Fig. 4 that relates to FPH processing. At this point the FPH may first change the structure of the item. At 420, the FPH extracts properties, and updates the item based on the properties[[,]]. At 430, the FPH marks modified parts of the item as promoted. After 430, the process proceeds back to 360 of Fig. 3 for further FPM processing. At 360, the FPM marks the item as not promotionStale. At 370, the FPM applies the changes to the structured object store. At 380, the FPM performs a commit/rollback transaction and closes a file handle.

Please replace the paragraph beginning at pg. 11, ln. 8 with the following paragraph:

At 520, the save function queries the store where the file-backed item resides (this could be on another machine in the case of a remote file) to check for the FPH registered for this file extension. This returns details about (e.g. the assembly name & version #) [[for]] the FPH that promoted the file. At 530, the update function loads the appropriate FPH (e.g. it may be registered in a Global Assembly Cache of the respective machine) based on the above information. At 540, the save function invokes method(s) on the FPH in order to perform demotion. The changed item also contains a record of the changes that were made to the Item. These changes are tracked by the structured store API.